Logotipo

El contenido generado por IA puede ser incorrecto.

MODAM Herramientas de Automatización de Despliegues

Maestría en Desarrollo y Operaciones de Software

**Actividad grupal: Empleo de Chef Workstation**

**Domínguez Urías, Didier**

**Lopez Flamenco, Gerardo**

**Nieto Rivera, Athena**

**Soto Audelo, Jorge**

**Trejo Figueroa, José**

Julio de 2025

**Tabla de contenido**

[Instalación y configuración de Chef Workstation 3](#_heading=h.n700hm8nfqyt)

[Actualización del sistema 3](#_heading=h.9ogs42ia7h9y)

[Descarga del instalador de Chef Workstation 3](#_heading=h.c60ibwtbwlz5)

[Instalación del paquete descargado 4](#_heading=h.y1e16rqkjs3b)

[Verificación de la instalación 5](#_heading=h.o3guv667fgve)

[Verificación del binario y estructura interna 5](#_heading=h.wkz6pw54e4ep)

[Generación de estructura base para trabajo con recetas 6](#_heading=h.rc7g38t5ngal)

[Cookbooks 7](#_heading=h.y2sfzwxubj0g)

[Cookbook 1: fastapi\_app 7](#_heading=h.50tmzw9hg8m0)

[Paso 1: Crear el cookbook 7](#_heading=h.7wjeveriun9v)

[Paso 2: Crear el archivo de plantilla main.py 8](#_heading=h.cwxn063grfx6)

[Paso 3: Escritura detallada de la receta default.rb 9](#_heading=h.g3vpaz10p01q)

[Paso 4: Ejecución del cookbook y validación del despliegue 12](#_heading=h.wirvquf7rl4q)

[Cookbook 2 15](#_heading=h.xwk7mm622zf3)

[Cookbook 3 15](#_heading=h.pxelmun2gcgv)

[Cookbook 4 15](#_heading=h.ctivm8b7nxuz)

[Cookbook 5 15](#_heading=h.7s48o08diszl)

[Tabla de valoración 16](#_heading=h.82f0i0nr7949)

[Reflexión sobre los desafíos enfrentados 16](#_heading=h.3jnr0whxa5gx)

[Conclusiones 17](#_heading=h.upqygyzcgzab)

[Recomendaciones 18](#_heading=h.k7qtyofkse)

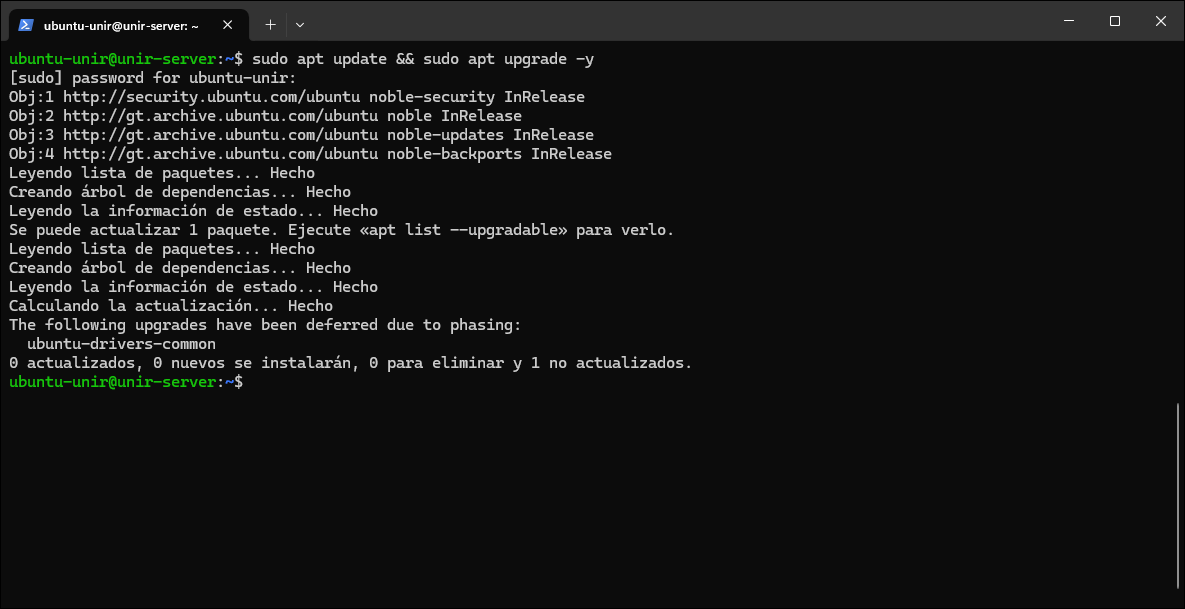
# Instalación y configuración de Chef Workstation

Para comenzar con la actividad, se procedió a realizar la instalación de Chef Workstation en un servidor con sistema operativo Ubuntu Server 24.04 LTS, siguiendo una serie de pasos sistemáticos que aseguran una instalación limpia, actualizada y funcional. A continuación, se detalla el proceso paso a paso:

## Actualización del sistema

Como buena práctica inicial, se actualizó el sistema para garantizar que todos los paquetes estén en su última versión disponible.

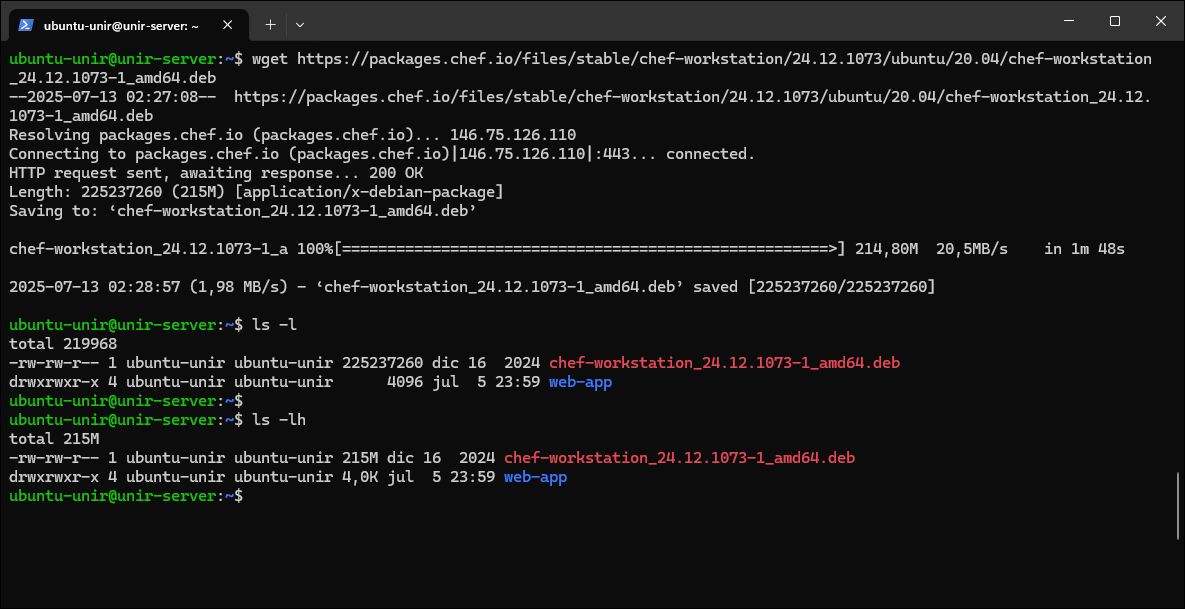
sudo apt update && sudo apt upgrade -y



## Descarga del instalador de Chef Workstation

Se descargó la versión Chef Workstation 24.12.1073, correspondiente a la LTS más reciente (julio 2025), desde el repositorio oficial de paquetes de Chef.

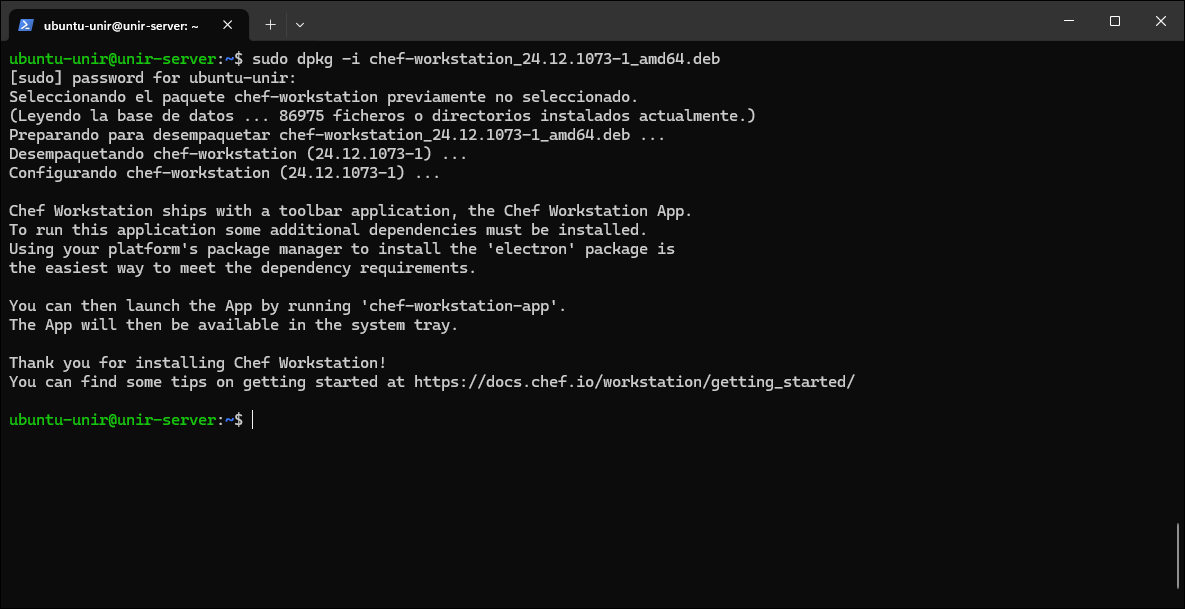
wget https://packages.chef.io/files/stable/chef- workstation/24.12.1073/ubuntu/20.04/chef-workstation\_24.12.1073-1\_amd64.deb



## Instalación del paquete descargado

Una vez completada la descarga, se procedió a instalar el paquete ***.deb*** utilizando ***dpkg***.

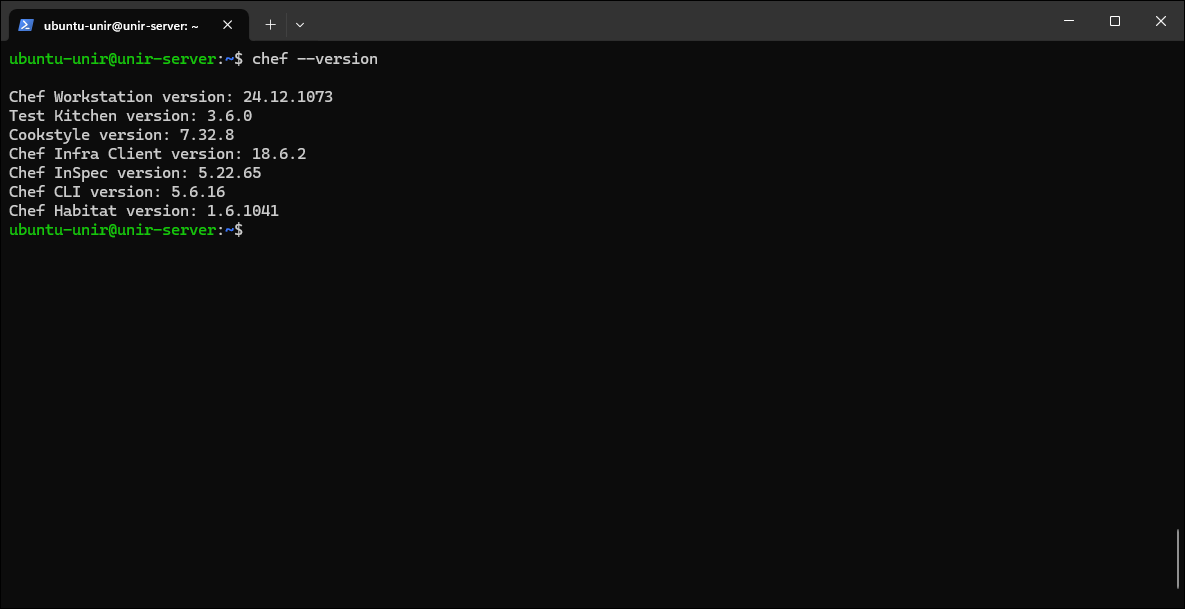
sudo dpkg -i chef-workstation\_24.12.1073-1\_amd64.deb



## Verificación de la instalación

Finalizada la instalación, se verificó que Chef Workstation se hubiera instalado correctamente, mediante el comando:

chef --version



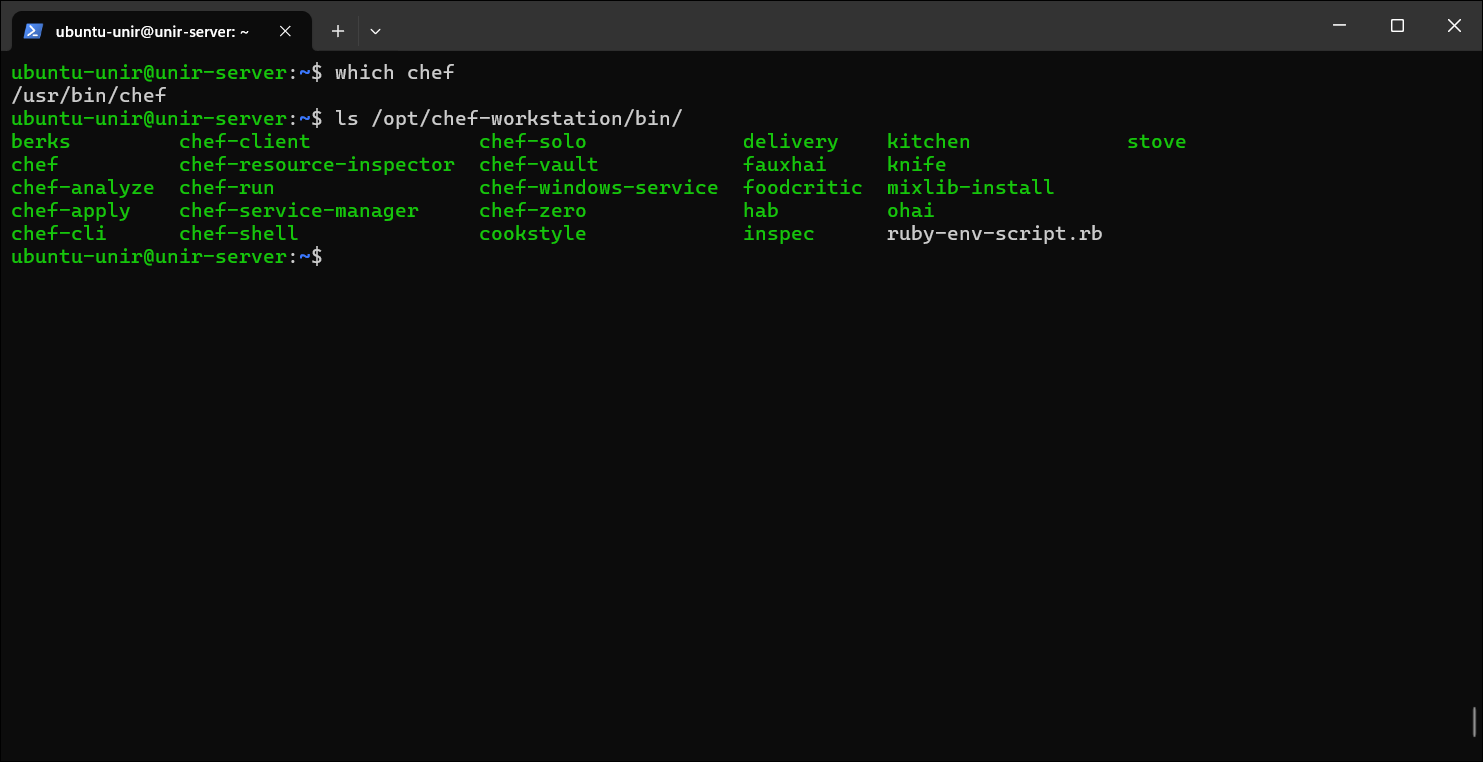
## Verificación del binario y estructura interna

Para asegurarse de que los ejecutables de Chef estaban correctamente ubicados, se ejecutaron los siguientes comandos:

which chef

ls /opt/chef-workstation/bin/

Este último comando listó todos los ejecutables incluidos con la instalación, como ***chef-client***, ***chef-run***, ***knife***, ***inspec***, ***kitchen***, entre otros.



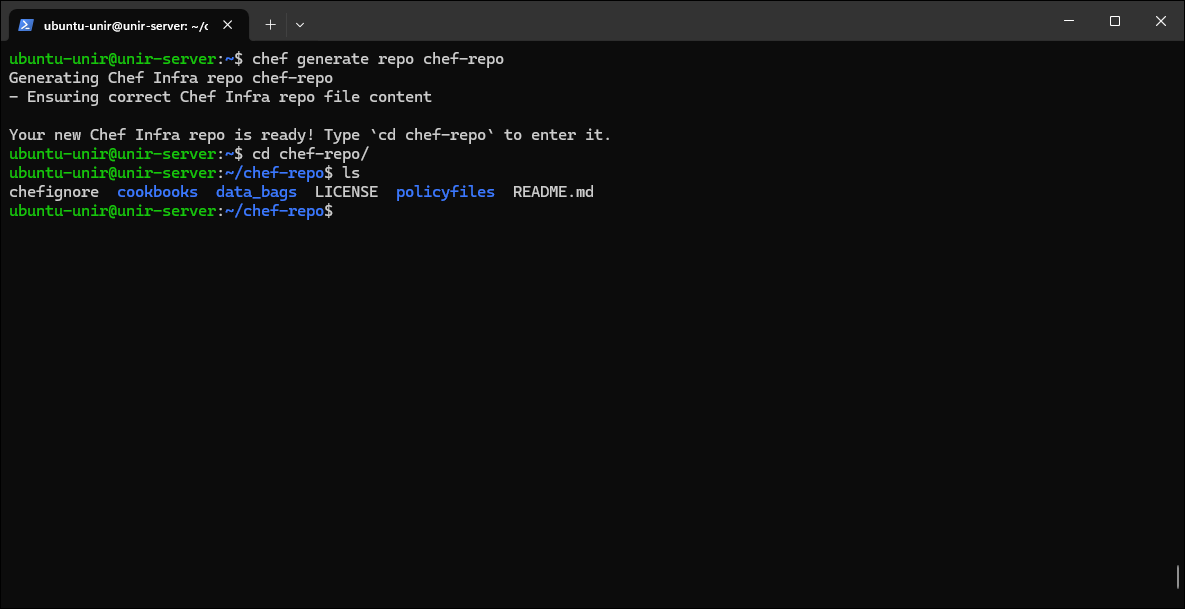
## Generación de estructura base para trabajo con recetas

Para asegurarse de que los ejecutables de Chef estaban correctamente ubicados, se ejecutaron los siguientes comandos:

chef generate repo chef-repo

cd chef-repo

Esto creó un directorio ***chef-repo*** con la estructura base que incluye carpetas para cookbooks, data bags y otras entidades propias del entorno Chef.



La instalación y configuración de Chef Workstation se realizó de manera exitosa. Todos los binarios fueron correctamente reconocidos, y se estableció el entorno base de trabajo para la posterior creación de recetas y cookbooks. Esta configuración servirá como punto de partida para las siguientes fases de la actividad.

# Cookbooks

## Cookbook 1: fastapi\_app

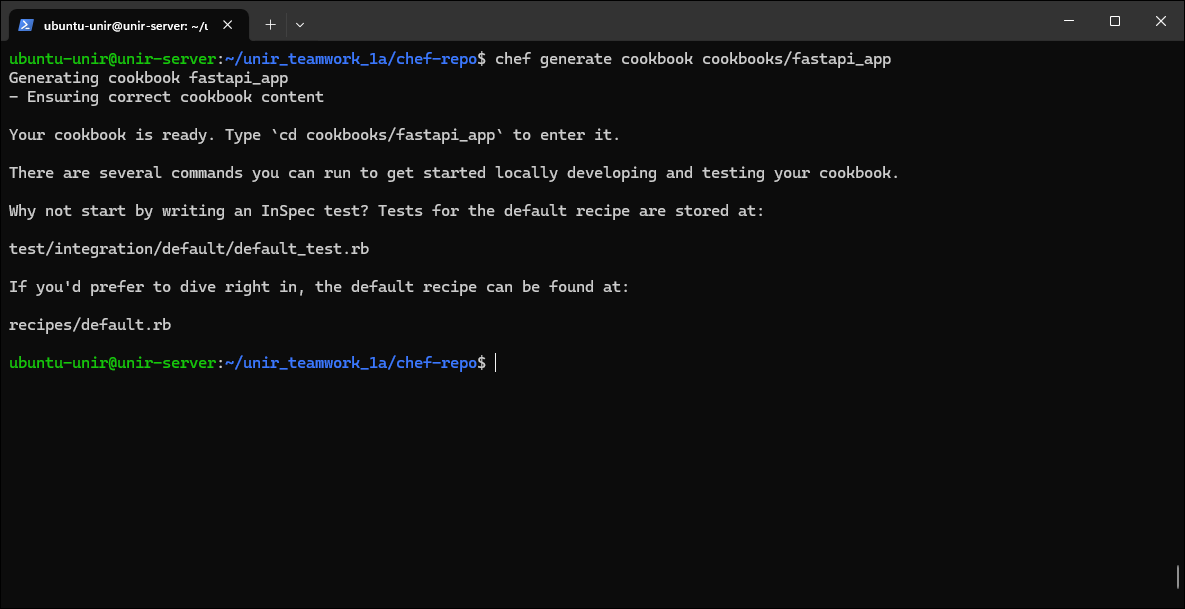
Cookbook personalizado llamado fastapi\_app con el objetivo de automatizar el despliegue de una aplicación web FastAPI sobre un servidor Ubuntu. El proceso incluyó:

* Instalación de herramientas necesarias.
* Creación de un entorno virtual de Python.
* Instalación de dependencias.
* Generación del código fuente (main.py) mediante plantilla.
* Ejecución automatizada de la aplicación mediante un servicio systemd.

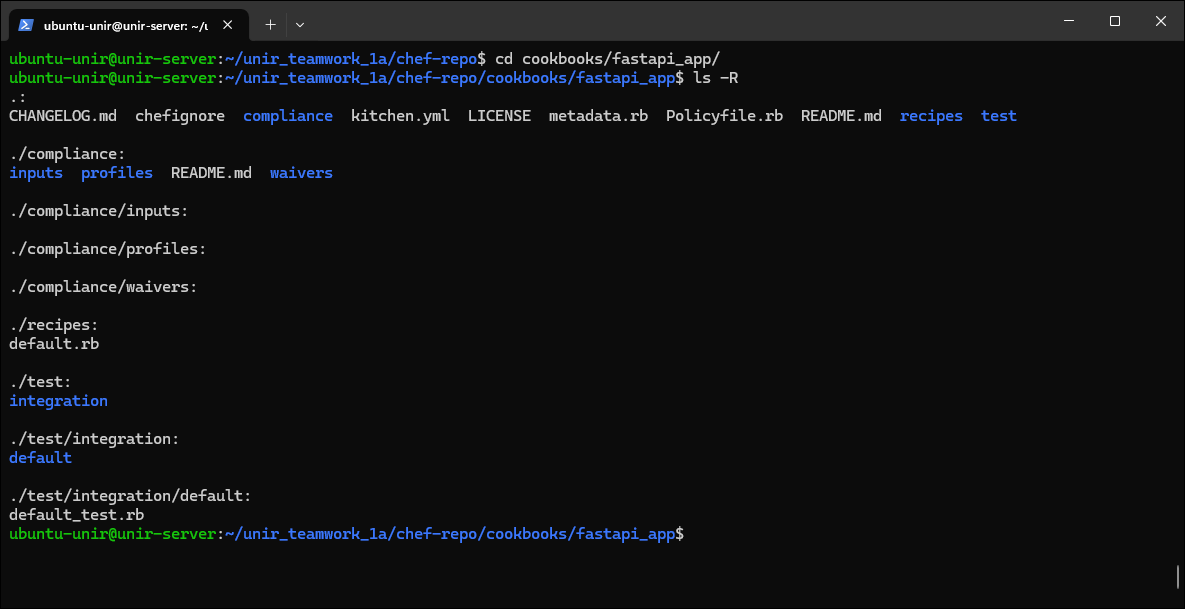
### Paso 1: Crear el cookbook

Desde la terminal, en el directorio ***chef-repo***, se ejecutó el siguiente comando:

chef generate cookbook cookbooks/fastapi\_app



Este comando generó la estructura básica del cookbook ***fastapi\_app***.



### Paso 2: Crear el archivo de plantilla main.py

Se creó el directorio de plantillas y el archivo:

mkdir -p cookbooks/fastapi\_app/templates/default

nano cookbooks/fastapi\_app/templates/default/main.py.erb

Con el siguiente contenido:

from fastapi import FastAPI

app = FastAPI(

*title*="Demo FastAPI for Chef Workstation",

*description*="API de demostración para la tarea del grupo 1\_A",

*version*="1.0.0"

)

@app.get("/")

*def* read\_root():

    return {

        "message": "Hello from FastAPI"

    }

@app.get("/healthcheck")

*def* healthcheck():

    return {

        "status": "ok"

    }

@app.get("/about")

*def* about():

    return {

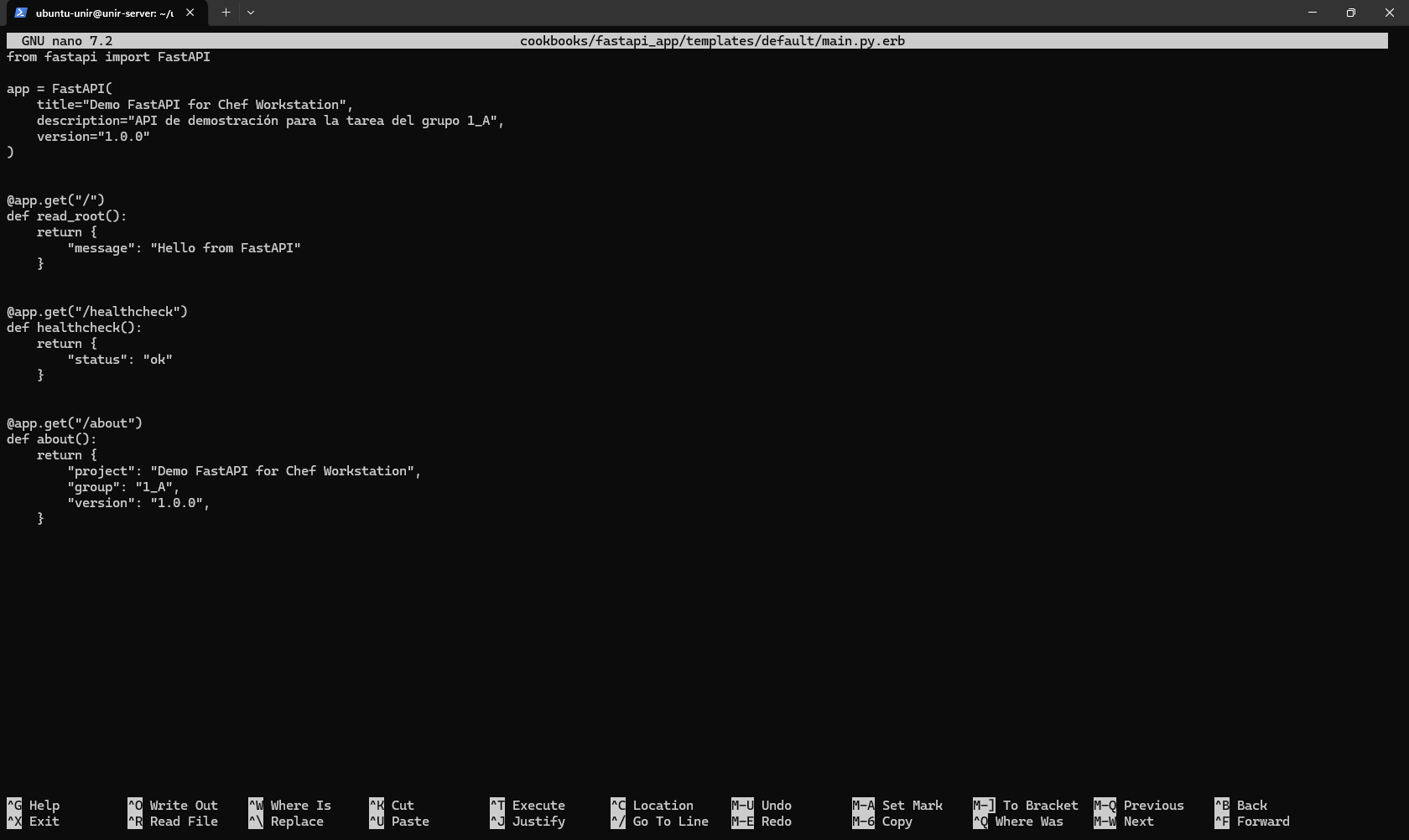
        "project": "Demo FastAPI for Chef Workstation",

        "group": "1\_A",

        "version": "1.0.0",

    }

Con el siguiente contenido:



### Paso 3: Escritura detallada de la receta default.rb

Con el cookbook ***fastapi\_app*** ya generado y la plantilla ***main.py.erb*** lista, se procedió a construir la receta principal ***default.rb***, ubicada en:

cookbooks/fastapi\_app/recipes/default.rb

Esta receta fue organizada en bloques funcionales claramente definidos, utilizando recursos nativos de Chef para asegurar una implementación idempotente, ordenada y reutilizable.

**Actualizar el sistema y preparar el entorno**

Se actualiza el sistema operativo para evitar conflictos con versiones obsoletas de paquetes y se instalan las herramientas necesarias:

execute 'apt\_update' do

  command 'apt update && apt upgrade -y'

end

package %w(python3-venv curl) do

  action :install

end

**Crear el directorio del proyecto**

directory '/home/ubuntu-unir/fastapi\_app' do

  owner 'ubuntu-unir'

  group 'ubuntu-unir'

  mode '0755'

  recursive true

  action :create

end

Este directorio contendrá todo el proyecto: entorno virtual, archivo de aplicación y scripts.

**Crear entorno virtual**

execute 'crear entorno virtual' do

  command 'python3 -m venv .venv'

  cwd '/home/ubuntu-unir/fastapi\_app'

  user 'ubuntu-unir'

  environment({ 'HOME' => '/home/ubuntu-unir' })

  not\_if {

::File.exist?('/home/ubuntu-unir/fastapi\_app/.venv/bin/activate')

}

end

Se crea el entorno solo si no existe, y se ejecuta como el usuario ***ubuntu-unir*** para evitar errores de permisos.

**Instalar FastAPI**

execute 'instalar fastapi' do

  command '/home/ubuntu-unir/fastapi\_app/.venv/bin/pip install "fastapi[standard]"'

  cwd '/home/ubuntu-unir/fastapi\_app'

  user 'ubuntu-unir'

  environment({

    'HOME' => '/home/ubuntu-unir',

    'PATH' => '/home/ubuntu-unir/fastapi\_app/.venv/bin:/usr/bin:/bin'

  })

end

Se instalan las dependencias en el entorno virtual, asegurando aislamiento del sistema principal

**Generar main.py desde plantilla**

template '/home/ubuntu-unir/fastapi\_app/main.py' do

  source 'main.py.erb'

  owner 'ubuntu-unir'

  group 'ubuntu-unir'

  mode '0644'

end

Este archivo define los endpoints ***/***, ***/healthcheck*** y ***/about*** de la aplicación FastAPI.

**Crear servicio systemd fastapi.service**

file '/etc/systemd/system/fastapi.service' do

  content <<~UNIT

    [Unit]

    Description=FastAPI Application

    After=network.target

    [Service]

    User=ubuntu-unir

    WorkingDirectory=/home/ubuntu-unir/fastapi\_app

    ExecStart=/home/ubuntu-unir/fastapi\_app/.venv/bin/python -m fastapi run main.py --host 0.0.0.0 --port 8080

    Restart=always

    [Install]

    WantedBy=multi-user.target

  UNIT

  mode '0644'

end

Este archivo define cómo se ejecutará la aplicación como servicio del sistema, de forma persistente y con reinicio automático en caso de falla.

**Recargar systemd y activar el servicio**

execute 'reload systemd' do

  command 'systemctl daemon-reexec && systemctl daemon-reload'

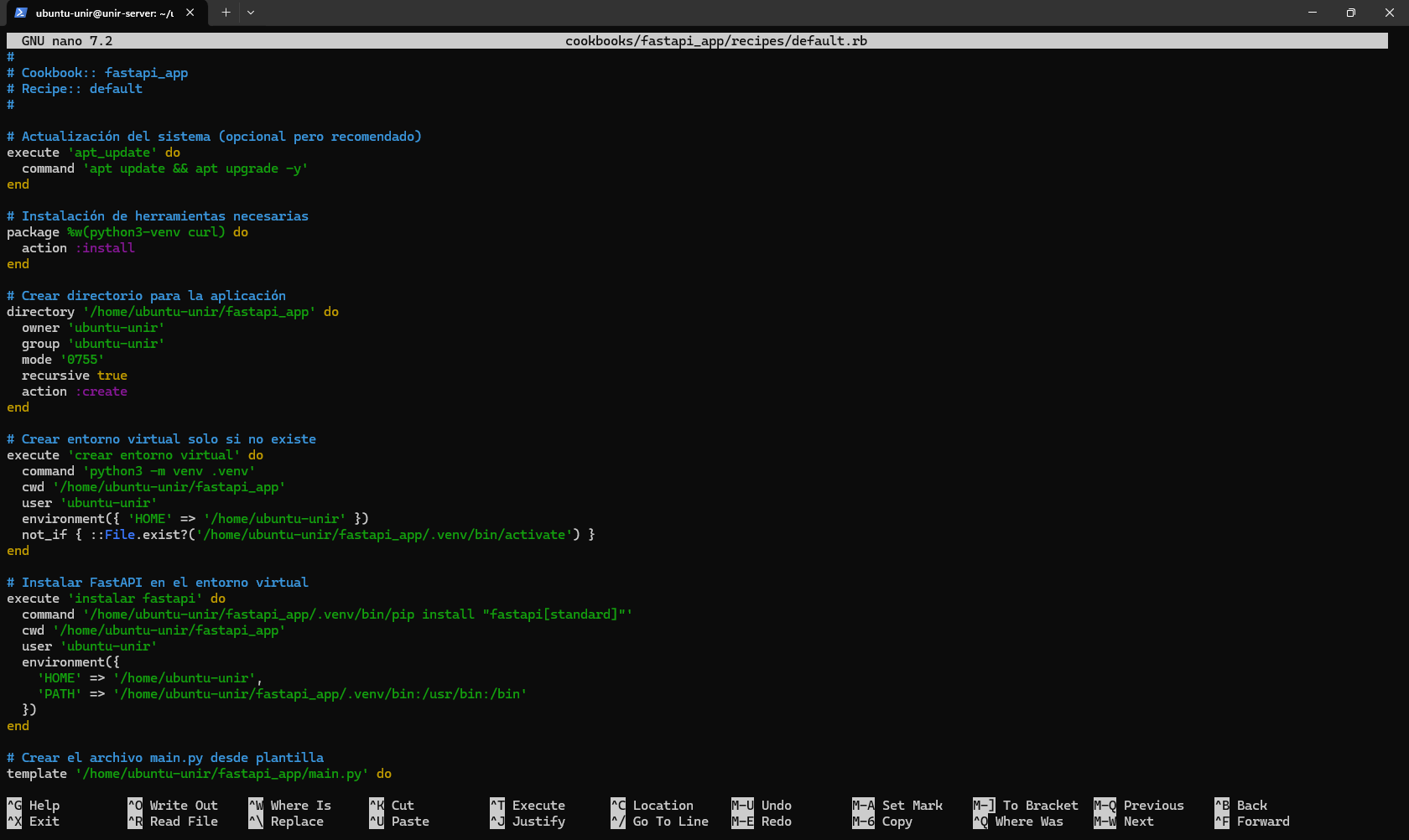
end

service 'fastapi' do

  action [:enable, :start]

end

Con esto, el servicio ***fastapi*** queda habilitado para iniciarse automáticamente con el sistema y se activa de inmediato.



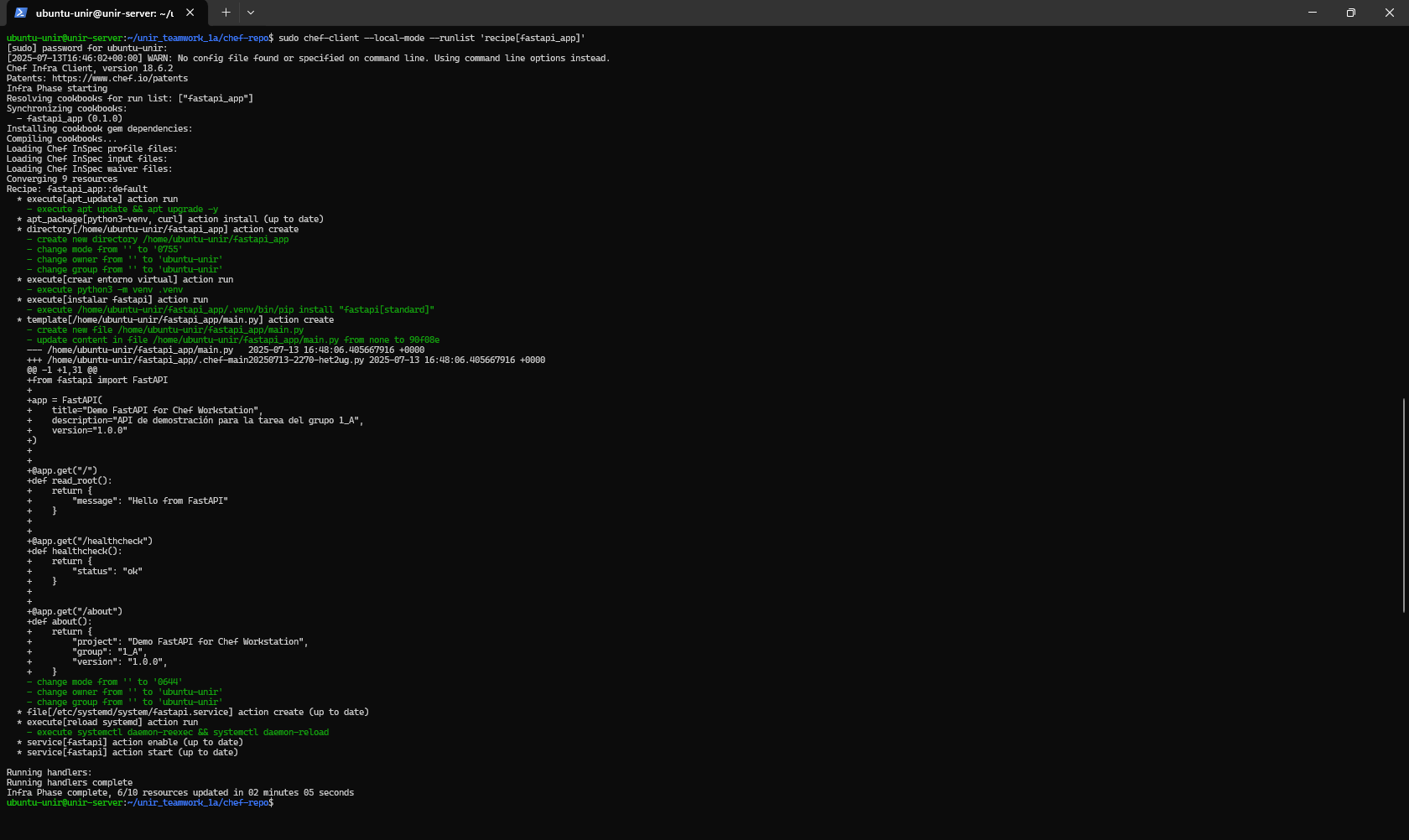
### Paso 4: Ejecución del cookbook y validación del despliegue

Una vez completada la receta ***default.rb***, se procedió a su ejecución para desplegar la aplicación FastAPI de forma automatizada.

**Ejecutar el cookbook con Chef**

sudo chef-client --local-mode --runlist 'recipe[fastapi\_app]'

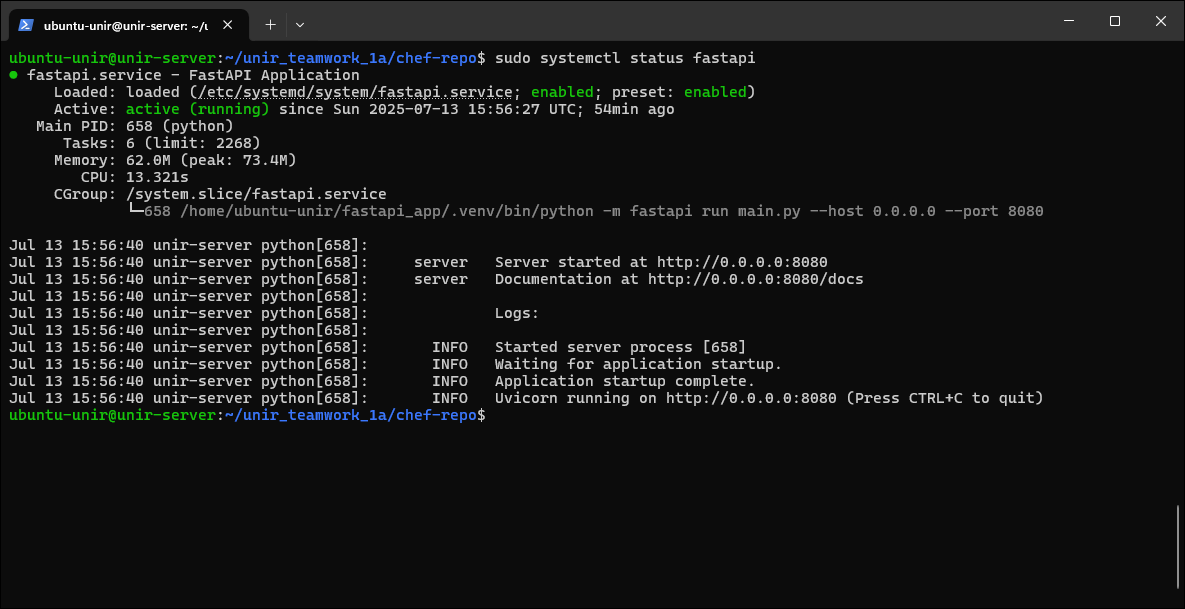
Este comando indica a Chef que ejecute en modo local (***--local-mode***) y aplique específicamente la receta fastapi\_app.



**Verificar que el servicio esté activo**

Después de aplicar la receta, se validó que el servicio fastapi estuviera activo con:

sudo systemctl status fastapi



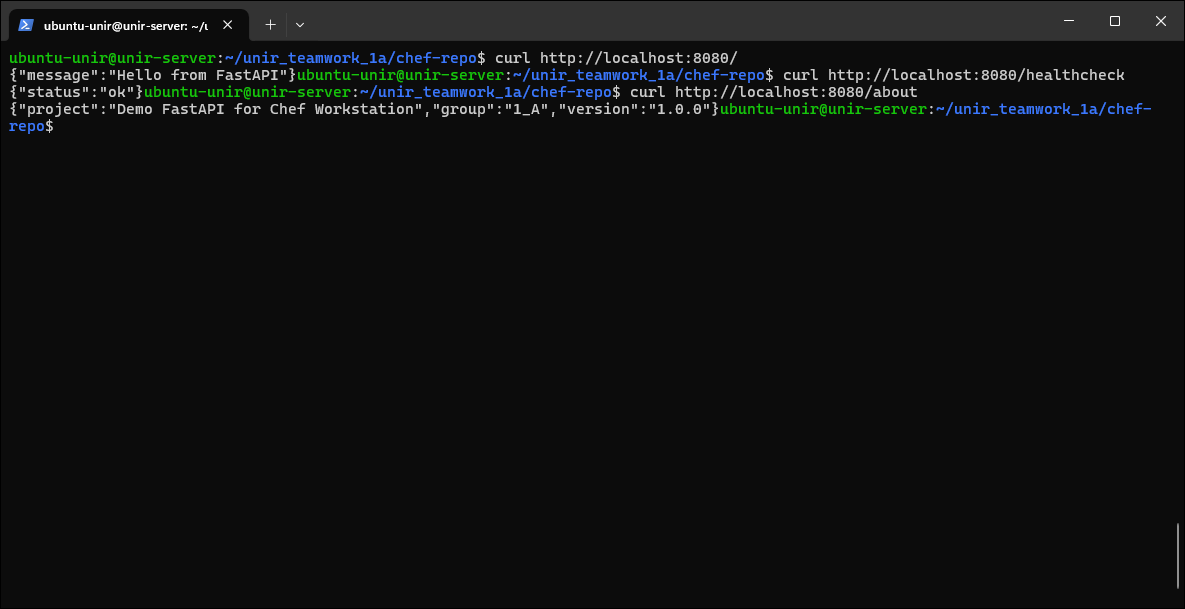
**Validar acceso a la API**

Se realizaron peticiones HTTP locales para confirmar que la aplicación FastAPI se encontraba en funcionamiento y respondiendo correctamente a sus rutas definidas.

curl http://localhost:8080/

curl http://localhost:8080/healthcheck

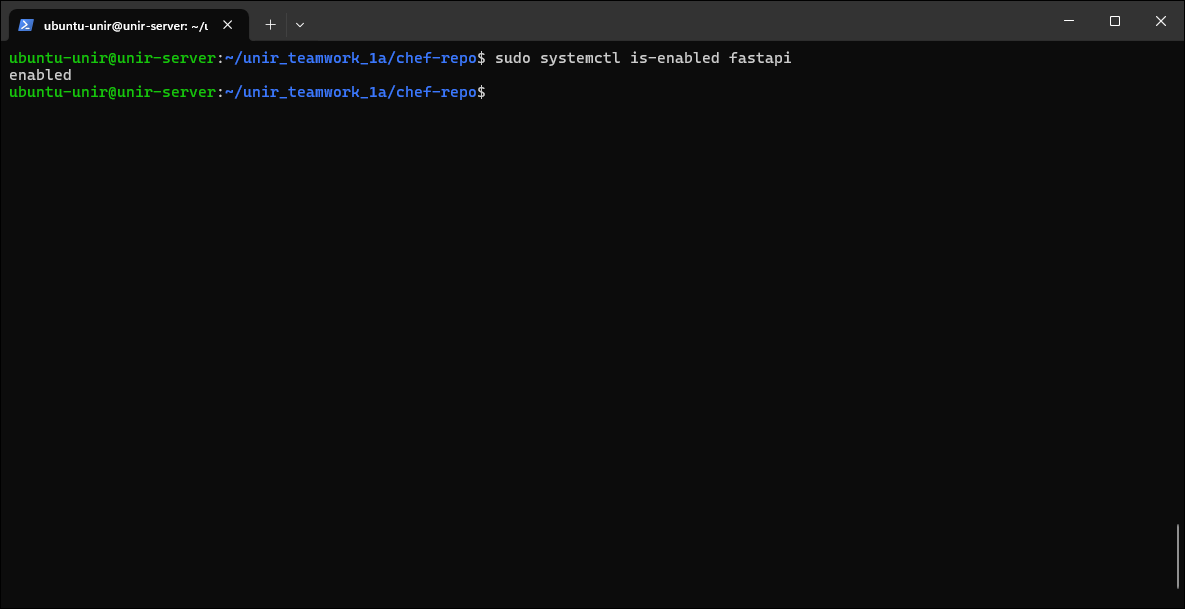
curl http://localhost:8080/about



**Verificar inicio automático del servicio**

Para confirmar que el servicio se inicia automáticamente con el sistema, se ejecutó:

sudo systemctl is-enabled fastapi



El despliegue fue exitoso. La aplicación FastAPI quedó:

* Ejecutándose en el puerto 8080.
* Expuesta como servicio systemd.
* Respondiendo a todas las rutas esperadas.
* Preparada para iniciar automáticamente al reiniciar el servidor.

Esto demuestra que la receta es funcional, reproducible y adecuada para producción o entornos educativos.

## Cookbook 2: RunSpringbootAPP

Cookbook personalizado llamado RunSpringbootApp con el objetivo de automatizar el despliegue de una aplicación web Java desde un repositorio remoto en un ambiente de MacOS. El proceso incluye:

* Definición de variables importantes.
* Instalación de dependencias usando brew.
* Clonado de un repositorio público.
* Construcción del JAR.
* Eliminar un proceso ya existente de java en caso de que lo haya.
* Ejecución automatizada de la aplicación mediante Java.

**Se ejecutò el siguiente comando para generar el cookbook**

chef generate cookbook cookbooks/RunSpringbootAppCookbook 3

## 

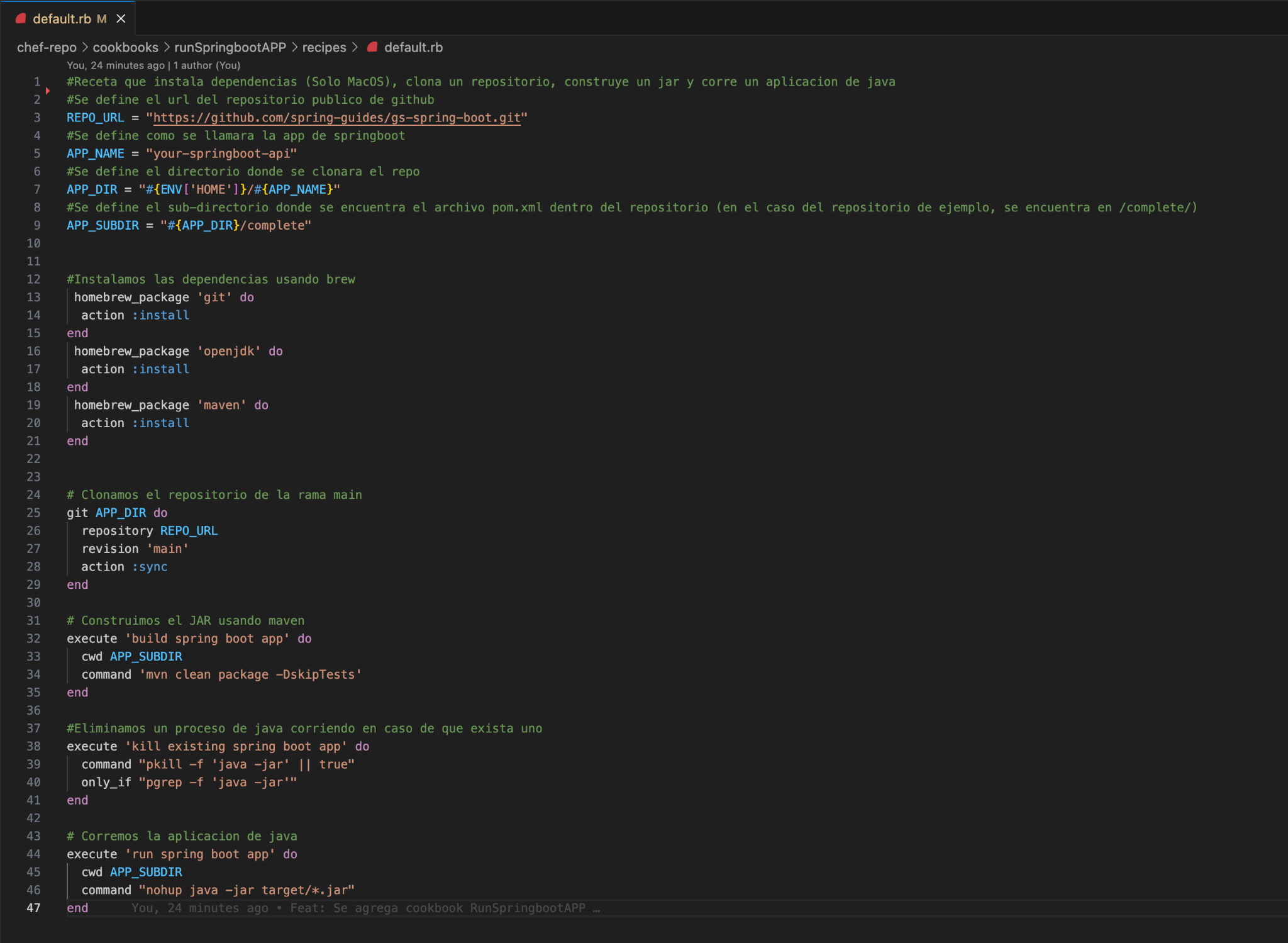
## 

## Paso 1: Se seleccionò el repositorio: <https://github.com/spring-guides/gs-spring-boot/tree/main>

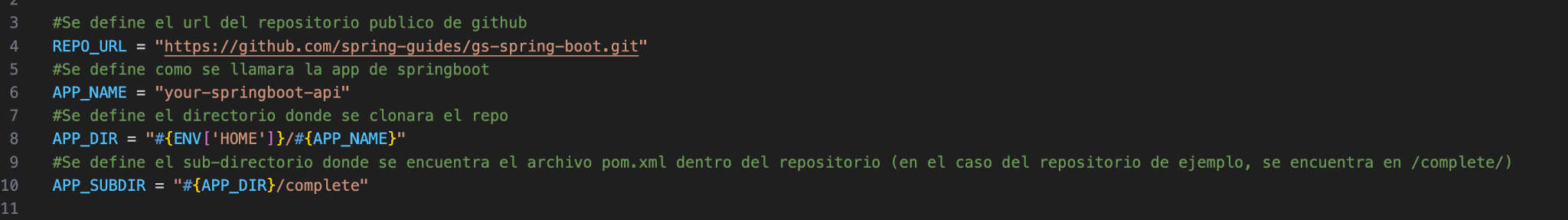
Como ejemplo para el cookbook, se utilizarà uno de los repositorios ejemplos oficiales de springboot, el cual simplemente inicia un servidor web en el puerto 8080, al enviar peticiones GET a la raiz del servidor, este te responde con "Greetings from Spring Boot!".

## 

**Paso 2: Se crea la receta en** [**default.rb**](http://default.rb)**:**

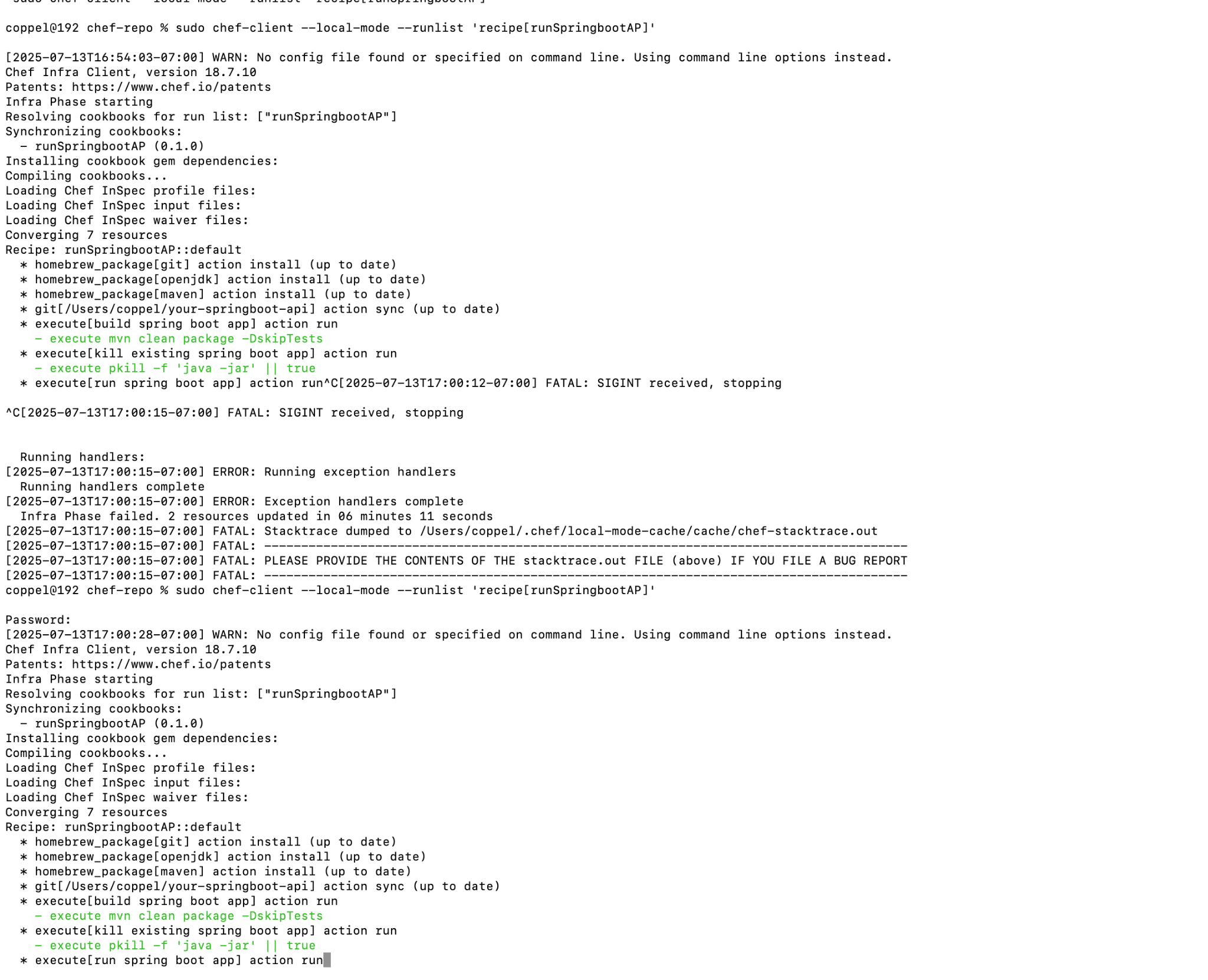
****

**Paso 3: Se definen las variables generales (su proposito viene comentado dentro de la receta)**

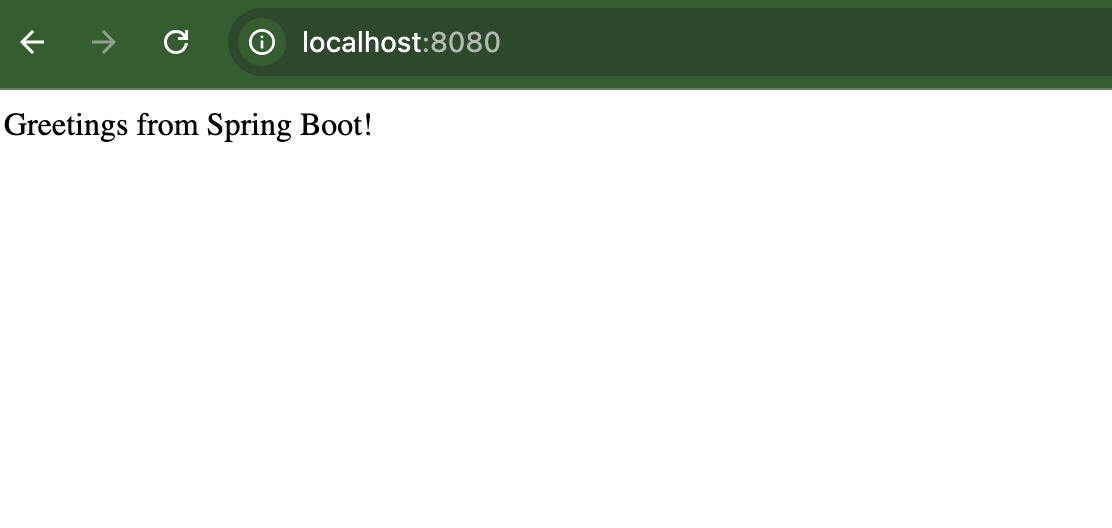
****

**Paso 4: Se ejecuta el cookbook con el siguiente comando**

sudo chef-client --local-mode --runlist 'recipe[RunSpringbootAPP]'

****

**Paso 5: Se accede al navegador al enlace: localhost:8080 para verificar el correcto funcionamiento del aplicativo**

****

## Cookbook 4

## Cookbook 5

# Tabla de valoración

|  | Sí | No | A veces |
| --- | --- | --- | --- |
| Todos los miembros se han integrado al trabajo del grupo |  |  |  |
| Todos los miembros participan activamente |  |  |  |
| Todos los miembros respetan otras ideas aportadas |  |  |  |
| Todos los miembros participan en la elaboración del informe |  |  |  |
| Me he preocupado por realizar un trabajo cooperativo con mis compañeros |  |  |  |
| Señala si consideras que algún aspecto del trabajo en grupo no ha sido adecuado |  |  |  |

# Reflexión sobre los desafíos enfrentados

Durante el desarrollo de la actividad, el equipo enfrentó varios desafíos técnicos y organizativos que requirieron investigación, comunicación y toma de decisiones colaborativa. A continuación, se enumeran los principales obstáculos encontrados:

* **Configuración inicial de Chef Workstation en Ubuntu Server:** Uno de los principales obstáculos fue la configuración correcta del entorno de Chef Workstation en Ubuntu Server, especialmente en lo relativo a la disponibilidad del comando chef, las rutas del binario y la necesidad de configurar correctamente el entorno de shell para evitar errores como command not found.
* **Problemas al intentar usar nohup desde el cookbook 1 (fastapi\_app):** Se descubrió que Chef no puede gestionar correctamente procesos que quedan en segundo plano con nohup, ya que este comando bloqueaba la ejecución de la receta. Se reemplazó por una solución más robusta con la creación de un servicio systemd, que permite iniciar, detener y monitorear la aplicación de forma segura.
* **Idempotencia de la receta Chef:** Algunos comandos se debieron condicionar con not\_if para evitar reejecuciones innecesarias y garantizar que la receta fuera idempotente (repetible sin efectos adversos).
* **Separación entre lógica de configuración y contenido dinámico:** Se aprendió a utilizar plantillas (template) para generar archivos como main.py, manteniendo la receta más clara y reutilizable, separando la configuración de los contenidos específicos de la aplicación.

# Conclusiones

* La herramienta Chef Workstation demostró ser poderosa y flexible para la automatización de despliegues, siempre que se estructure correctamente la lógica en cookbooks bien definidos.
* Es posible automatizar no solo la instalación de paquetes, sino también el despliegue completo de aplicaciones modernas, incluyendo su ejecución como servicios del sistema.
* El enfoque modular y declarativo de Chef favorece la reproducibilidad, una característica clave para entornos DevOps, pruebas e infraestructura como código.
* La experiencia permitió al equipo desarrollar habilidades técnicas prácticas, trabajar en equipo y documentar un proceso técnico completo.

# Recomendaciones

* Para futuros equipos, se recomienda probar primero todos los pasos manualmente, antes de codificarlos en una receta. Esto ayuda a entender mejor los comandos, rutas y posibles errores.
* Evitar el uso de comandos como sudo dentro de las recetas Chef, ya que Chef se ejecuta como root.
* Siempre validar rutas, usuarios y permisos cuando se utilizan servicios como systemd.
* Aprovechar los recursos oficiales de documentación de Chef y los foros de la comunidad cuando surgen problemas específicos.
* Tomar capturas de pantalla durante cada etapa para no olvidar pasos importantes y facilitar la documentación.